CSC 364H (L0101) **Test 1** **Summer 99**

Name ______________________________ Student No. ______________________________

No aids allowed. Answer all questions on test paper. Use backs of sheets for scratch work.

Total Marks: 30

[6] 1. The Simple Knapsack *Search* problem takes as input the numbers:

$$d, w_1, w_2, \ldots, w_d, C \in \mathbb{N}$$

and requires as output a set $S_0 \subseteq \{1, \ldots, d\}$ such that $K(S_0) = M$, where

$$M = \max_{S \subseteq \{1,\ldots,d\}} \{K(S) | K(S) \leq C\}$$

and $K(S) = \sum_{i \in S} w_i$. Recall that the greedy method for SKS produces an answer $\overline{M}$ such that $\overline{M} \leq C$ but possibly $\overline{M} < M$.

Lemma: Either $\overline{M} = M$, or $\overline{M} > \frac{1}{2}C$.

Your task is to prove this lemma by following the outline below:

Proof of Lemma: Assume that $\overline{M} \neq M$ and $\overline{M} \leq \frac{1}{2}C$, in order to get a contradiction. Now, let S_0 be the result of greedy, i.e. a subset of $\{1, \ldots, d\}$ such that $K(S_0) = \overline{M}$.

(a) Show that there is at least one weight in S_0:

(b) Show that there is at least one weight not in S_0:

(c) Show now that $\overline{M} > \frac{1}{2}C$ (and hence get the contradiction) by considering the first weight, call it w_j, which has been rejected after at least one weight has been added:

[12] 2. Recall the Longest Monotone Subsequence (LMS) Problem:

Input: $d, a_1, a_2, \ldots, a_d \in \mathbb{N}$

Problem: Find L = length of the longest monotone non-decreasing subsequence.

Note: the subsequence need not be consecutive. Thus $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ is a monotone subsequence provided $1 \leq i_1 < i_2 < \ldots < i_k \leq d$ and $a_{i_1} \leq a_{i_2} \leq \ldots \leq a_{i_k}$. For example, the length of the longest monotone subsequence of $\{4, 6, 5, 9, 1\}$ is 3, and one such subsequence is given by $\{4, 5, 9\}$.

Give a dynamic programming solution to the LMS Problem:

(a) Define an array of subproblems:

(b) Find a recurrence (no justification necessary):

(c) Write an algorithm:

[12] 3. The Activity Selection with Profits problem is defined as follows:

Input: A list of activities $(s_1, f_1, p_1), \ldots, (s_n, f_n, p_n)$, where each activity consists of a start time s_i and a finish time f_i (such that $0 \leq s_i < f_i$), and a profit $p_i > 0$ obtained if the activity is selected.

Problem: Find a set $S \subseteq \{1, \ldots, n\}$ of selected activities such that no two activities in S are in conflict and $\sum_{i \in S} p_i$ is maximal.

The dynamic programming solution that was given in lectures involved computing an array $A[j]$ whose value is the maximum profit that can be obtained by selecting a set of activities that completes by time u_j (where $u_1 < u_2 < \ldots < u_k$ is the list of all the distinct finish times, sorted in increasing order).

Assuming that the array just described has already been evaluated, show how to use the values in the array to compute the actual set S of selected activities:

Hint: consider what happens if $A(k) = A(k-1)$, and what happens if $A(k) > A(k-1)$, and then give a general procedure.